UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

MAESTRÍA EN TELECOMUNICACIONES

TEMA:

"SISTEMA DE RESPUESTA DE VOZ INTERACTIVA BASADO EN FREESWITCH PARA CENTROS DE INFORMACIÓN AUTOMATIZADOS"

AUTOR:

DOMINGUEZ DIAZ GUIDO DANIEL

Trabajo de titulación previo a la obtención del grado de

Magister en Telecomunicaciones

TUTOR:

MSc. MANUEL ROMERO PAZ

Guayaquil, a los 21 días del mes de agosto del año 2017

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

MAESTRÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por Domínguez Díaz Guido Daniel como requerimiento parcial para la obtención del Título de Magíster en Telecomunicaciones.

TUTOR

_____

MSc. Manuel Romero Paz

DIRECTOR DEL PROGRAMA

_____

MSc. Manuel Romero Paz

Guayaquil, a los 21 días del mes de agosto del año 2017

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO
MAESTRÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

YO,

DECLARO QUE:

El trabajo de Titulación "**Sistema de respuesta de voz interactiva basado en FREESWITCH para centros de informatización automatizados.**" previa a la obtención del Título de **Magíster en Telecomunicaciones**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 21 días del mes de agosto del año 2017

EL AUTOR

_____

Domínguez Díaz Guido Daniel

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO

MAESTRÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, Domínguez Díaz Guido Daniel

Autorizo a la Universidad Católica de Santiago de Guayaquil a la **publicación**, en la biblioteca de la institución del Trabajo de Titulación, "**Sistema de respuesta de voz interactiva basado en FREESWITCH para centros de informatización automatizados**", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

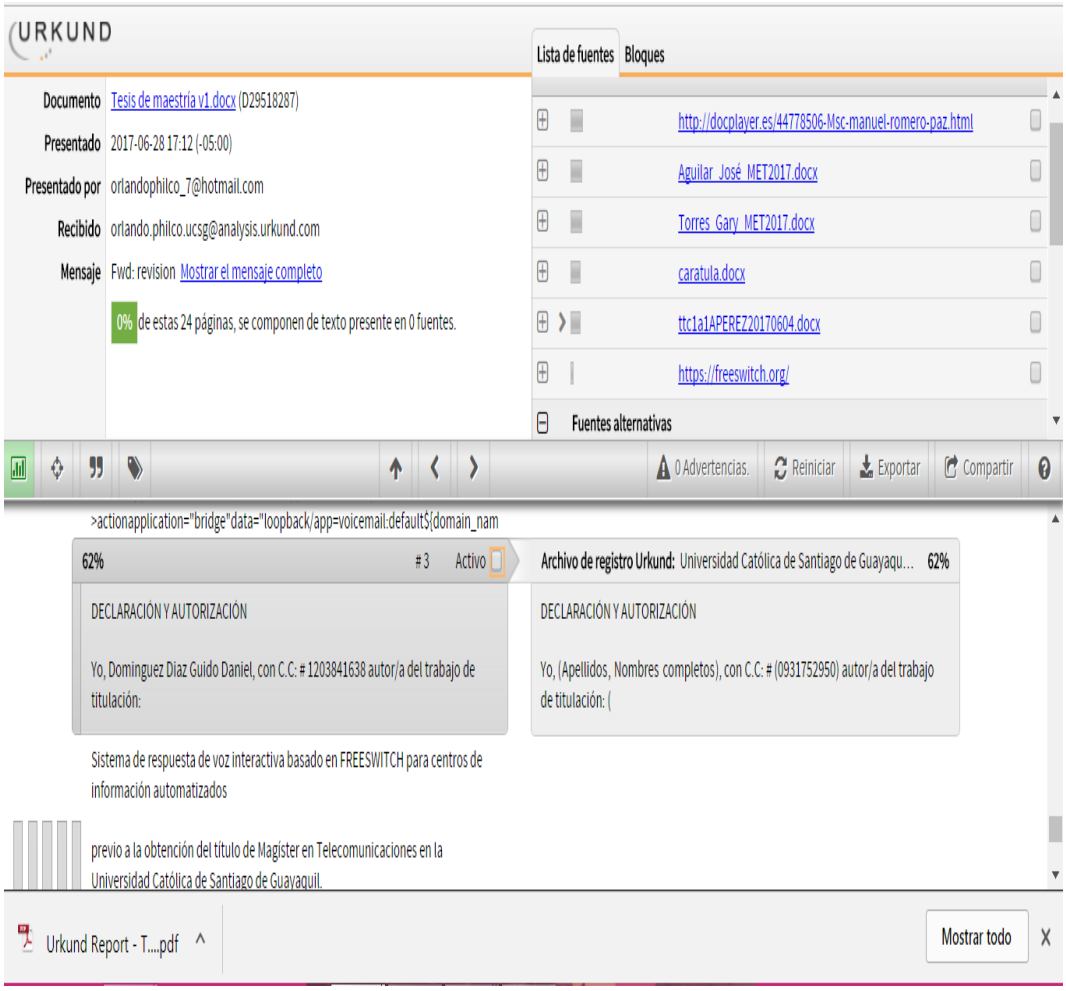
Guayaquil, a los 21 días del mes de agosto del año 2017

EL AUTOR

_____

Domínguez Díaz Guido Daniel

REPORTE URKUND



Dedicatoria

Este Trabajo de Titulación va dedicado a mi padre el MSc. Guido Dominguez, que a pesar de ya no estar con nosotros, su actitud ante las dificultades y su determinación para vencerlas me marco e inspiró profundamente.

Estoy seguro que desde el cielo, me ayudará a tomar buenas decisiones por el bien común de la sociedad.

Agradecimientos

A Dios porque me ha dado la oportunidad de contar con magnificas personas que han aportado conocimiento, amor, comprensión y felicidad a mi vida.

A mis padres porque no escatimaron esfuerzos al darme la facilidad necesaria para cumplir con las metas académicas trazadas.

A mi esposa Yasmin por brindarme su apoyo total durante el tiempo empleado en estos estudios.

A todos los profesores de la Maestría en Telecomunicaciones de la Universidad Católica de Guayaquil, y en especial a mi tutor el MSc. Manuel Romero Paz por guiarme con sus conocimientos y consejos para la culminación de este trabajo.

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

SISTEMA DE POSGRADO
MAESTRÍA EN TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

f. _____

MSc. Manuel Romero Paz
TUTOR

f. _____

MSc. Manuel Romero Paz
DIRECTOR DEL PROGRAMA

f. _____

MSc. Luis Córdova Rivadeneira
REVISOR

f. _____

MSc. Orlando Philco Asqui
REVISOR

RESUMEN

En este trabajo se presenta una propuesta de un sistema de Respuesta de Voz Interactiva (IVR) usando *software* libre para la implementación de centros de información automatizados en beneficio del proceso de informatización de la sociedad. Se describen las generalidades de la integración computadora teléfono (CTI), los *softwares* existentes que permiten la creación de sistemas IVR y se fundamenta el uso del FreeSWITCH como mejor alternativa. Se explican las distintas configuraciones realizadas en el FreeSWITCH para el correcto funcionamiento del menú IVR y otras aplicaciones requeridas. Se exponen los criterios que avalan la selección del *software* X-Lite como cliente. Se seleccionan las tarjetas de interfaz con la línea telefónica más apropiadas para la aplicación considerando costo y máxima compatibilidad con FreeSWITCH. Finalmente, se muestra el diseño de la aplicación de control del sistema, implementada con el Entorno de Desarrollo Integrado (IDE) Visual Studio 2015, describiéndose sus funcionalidades a través de la presentación de la interfaz gráfica.

Palabras clave: IVR, CTI, FreeSWITCH, X-Lite, IDE

ABSTRACT

This paper presents a proposal for an Interactive Voice Response system using free software for the implementation of automated information centers for the benefit of the computerization of society. It describes the generalities of computer telephone integration (CTI), the existing software that allow the creation of IVR systems and select FreeSWITCH as a better alternative. The different configurations made in the FreeSWITCH are explained for the correct operation of the IVR menu and other required applications. The criteria that support the selection of the X-Lite software as a client are set out. The interface cards with the most appropriate telephone line for the application are selected considering cost and maximum compatibility with FreeSWITCH. Finally it shows the design of the system control application implemented with the Integrated Development Environment (IDE) Visual Studio 2015, describing its functionalities through the presentation of the graphical interface.

Keywords: *IVR, CTI, FreeSWITCH, X-Lite, IDE.*

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 2

CAPÍTULO 3

DESCRIPCIÓN DEL PROYECTO DE INTERVENCIÓN.

Introducción

El acelerado desarrollo de las tecnologías de la informática y las comunicaciones ha propiciado la convergencia de los servicios de voz y datos en una misma aplicación, con el objetivo de acceder a la información de la forma más simple posible y al alcance de todos. La integración de los servicios automatizados es la tendencia en la que se mueven los sistemas de telecomunicaciones y de computación, esto ha conllevado a una estrecha interrelación entre ellos.

En este trabajo se realiza una propuesta de un Sistema de Respuesta de Voz Interactiva basado en FreeSWITCH para la creación de centros de información automatizados tecnológicamente independientes en beneficio de la sociedad.

A continuación, se justificará y delimitará el problema a investigar, se establecerán los objetivos generales y específicos, la hipótesis y la metodología de la investigación.

Justificación del tema

Los sistemas de Respuesta de Voz Interactiva o IVR (del inglés *Interactive Voice Response*) permiten al ciudadano interactuar de forma remota y automática con el centro del que desea obtener información, sin que sea necesario que exista una persona física al otro lado de la comunicación telefónica. Para ello, el usuario marca el número de teléfono del organismo e interactúa con el menú de voz que se le presente, por vía palabras (pues la mayor parte de ellos implementan reconocimiento fonético) o por vía marcación de teclas en el terminal telefónico, el sistema busca la información demandada en las bases de datos del centro y se la proporciona al usuario a través de un mensaje de voz.

Las principales ventajas del uso de estos sistemas y que dan lugar a esta investigación son: la disminución del tiempo de atención al usuario, el uso

eficiente de las bases de datos y una mayor disponibilidad de la información. Implementar este sistema con el uso del motor de telefonía FreeSWITCH presupone otra gran ventaja para este proyecto: el uso de *software* libre.

Antecedentes

En la década de 1970 la tecnología IVR experimenta un gran crecimiento, aunque se consideraba compleja y costosa para la automatización de tareas en los centros de llamadas. Los primeros sistemas de respuesta de voz se basaban en la tecnología DSP (del inglés de *Digital Signal Processing)* y estaban limitados a vocabularios reducidos.

A principios de 1980, *Perception Technology* se convirtió en el principal competidor del mercado, después de que la tecnología de discos duros alcanzara precios rentables. En aquel momento, un sistema podía guardar palabras digitalizadas en un disco, reproducir el mensaje hablado apropiadamente y procesar la marcación DTMF (del inglés *Dual-Tone Multi-Frequency*).

En la actualidad, con el progreso de las Tecnologías de la Información y las Comunicaciones, los sistemas IVR brindan al usuario un menú dinámico en el cual se pueden reproducir anuncios pregrabados, se hace reconocimiento de voz, se interpretan los dígitos marcados y se accede a bases de datos, con la finalidad de satisfacer la necesidad de información del usuario.

La respuesta de voz interactiva se puede utilizar para establecer la interfaz inicial de las operaciones de un centro cualquiera, dígase empresas, hospitales, cines, agencias, etcétera. Una vez identificadas las solicitudes del usuario se puede brindar respuestas a preguntas como, saldos de cuentas, estado de salud de un paciente, estado de un proceso o gestión, entre otras, sin necesidad de que intervenga un operador telefónico. En caso de que el sistema desconozca la información solicitada es capaz de redirigir la llamada a un agente del centro con un conjunto de aptitudes específicas.

Definición del problema

En la actualidad para implementar muchos de los sistemas de Respuesta de Voz Interactiva se requiere del uso de programas que necesitan licencias y no son flexibles en sus configuraciones, es por esto que surge la necesidad de crear centros de información automatizados usando tecnología de *software* libre como una vía eficiente y económica de acceso a información contenida en bases de datos.

Objetivo general

Diseñar un *software* cliente-servidor para implementar un sistema IVR para centros de información automatizados.

Objetivos específicos

- Caracterizar los sistemas IVR haciendo énfasis en la función del *software* cliente-servidor.
- Describir las características de los *softwares* Asterisk y FreeSWITCH, para evaluar cuál de ellos es adecuado para la aplicación.
- Evaluar cuál de los *softwares* clientes existentes actualmente es el más adecuado y proponer la configuración para su utilización.
- Implementar un *software* servidor utilizando el IDE Visual Studio 2015 que permita controlar la tarjeta de interfaz con la línea telefónica existente y realizar la configuración de los parámetros del FreeSWITCH para su funcionamiento como centro de información automatizado.

Hipótesis

Si se propone un sistema IVR usando *software* libre se podrán implementar centros de información automatizados con tecnología propia para satisfacer de forma más eficiente la necesidad de información de la sociedad.

Metodología de la investigación

- Método de observación documental y científica: Se emplea con el objetivo de obtener información y lograr la definición del problema del marco teórico y el desarrollo de la tesis.
- Método analítico: Se emplea con objetivo de analizar los elementos de forma separada para ver las relaciones entre ellos.

CAPITULO 1 . Integración computadora teléfono y sistemas IVR.

La Integración Computadora Teléfono (Computer Telephony Integration, CTI) se define como la base tecnológica que permite el enlace entre el sistema de computadoras y el sistema telefónico para crear aplicaciones que integran funciones tanto del mundo del teléfono como de la computación (Santos, 2006).

Existen dos tipos de aplicaciones CTI, las de primer nivel y las de tercer nivel. La posición en que se encuentra la conexión de la computadora con relación a la línea telefónica define el nivel.

1.1 Aplicaciones CTI de primer nivel.

La característica principal de un sistema CTI de primer nivel es el empleo de un dispositivo que se encarga de las acciones de procesamiento de las llamadas, interrumpiendo la señalización existente entre la central de conmutación y el terminal telefónico. Este tipo de sistema no requiere que la computadora esté conectada en una red de datos dado que no existe un servidor dedicado a establecer la comunicación entre la red pública y la de computadoras (Guillan, 1999).

Los componentes de la arquitectura de este tipo de sistema CTI son:

- Conmutador perteneciente a la Red de Telefonía Pública Conmutada.
- La computadora.
- Programa de aplicación en la computadora.
- Teléfono.

Como se mencionó anteriormente la computadora puede o no pertenecer a una red de datos, en caso de pertenecer, cada una de ellas necesita implementar una aplicación CTI de primer nivel. Como es característico de este tipo de aplicaciones, el enlace existente entre la computadora y la red de telefonía es similar al empleado por un terminal telefónico convencional y, por tanto, la

señalización es prácticamente la misma, con la excepción de sistemas con teléfonos complejos cuya señalización es propia de cada terminal. La intercepción de la señalización se realiza a través de una tarjeta que se acopla a la computadora por alguna de sus interfaces.

1.2 Aplicaciones CTI de tercer nivel.

En este caso el servidor de la red de computadoras conoce y maneja información de usuario y tiene una vista amplia del mundo telefónico, lo que le permite en principio asociar cualquier teléfono a cualquiera de sus computadoras. El servidor establece las llamadas por otras personas, retirándose cuando ésta se establece satisfactoriamente, maneja información sobre el progreso de la llamada, detalles de su configuración e información del solicitante y el solicitado. Puede, incluso, tratar con dos o más llamadas simultáneas.

En esta arquitectura existen al menos tres programas de aplicación, uno en el procesador del conmutador perteneciente a la Red de Telefonía Pública Conmutada, otro en el servidor de la red de computadoras y otro que es el *software* cliente instalado en una o en varias de las computadoras conectadas a la red. El *software* cliente es el único con el que interactúa el usuario por lo que generalmente está elaborado en un lenguaje de alto nivel con una interfaz interactiva caracterizada por un menú de opciones gráficas.

En esencia, los componentes de la arquitectura básica de CTI de tercer nivel coinciden con los del primer nivel. La diferencia fundamental radica en que el conmutador posee inteligencia propia y emplea protocolos para la comunicación entre el procesador del conmutador y el servidor de la red de computadoras. En la figura 1.1 se muestra el esquema de este tipo de aplicación CTI.

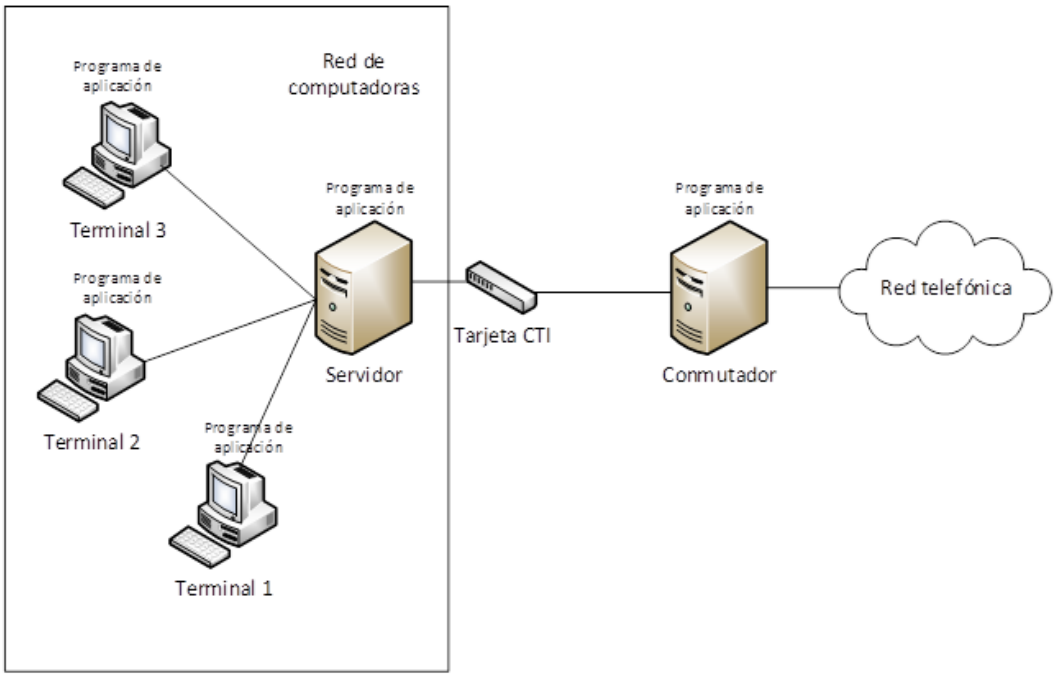


Figura 1.1. Esquema de aplicación CTI de tercer nivel.

Elaborado por el autor.

Los componentes de esta aplicación son:

- El teléfono.
- La red de computadoras.
- Programas de aplicación.
- Conmutador de la PSTN (Public Switched Telephone Network).
- Los protocolos.
- El servidor.

Estrechamente con este tipo de aplicación CTI se encuentran ligados los sistemas de respuesta de voz interactiva. Estos son sistemas muy utilizados en el mundo para atender a abonados solicitantes de información de cualquier empresa o centro público.

1.2.1 Respuesta de voz interactiva (IVR).

El auge de CTI ha dado paso al desarrollo de sistemas de IVR, los que han brindado un conjunto de nuevas posibilidades a la primera, de manera que en la actualidad ambas tecnologías aparecen juntas en la mayoría de las aplicaciones.

La IVR es una tecnología que permite a miles de empresas en el mundo atender llamadas telefónicas de manera automática, consultar o manipular bases de datos y proporcionar la información en forma de voz, pues permite que la información que se encuentra en sus servidores se encuentre disponible para los usuarios telefónicos que la requieran. Los centros de respuesta de llamada ven en los sistemas de IVR la manera de reducir los costos de atención al cliente, mejorar su satisfacción y automatizar los procesos comerciales.

Su funcionamiento se basa en responder a tonos o comandos de voz, obtener la información de su base de datos y proporcionarla al usuario en forma de voz. Los primeros sistemas de IVR simplemente usaban voz previamente pregrabada y los consumidores alcanzaban la opción que estaban buscando a través de los menús codificados.

Aunque es una manera de automatizar la contestación de la llamada, tiene sus limitaciones, incluyendo la frustración ocasional del usuario. En los sistemas modernos de IVR se ha mejorado la forma de contestación al usuario, esta puede ser por sonidos pregrabados o empleando la tecnología TTS (*Text To Speech*) de conversión de texto a voz. Los clientes pueden formular sus peticiones haciendo uso de los tonos DTMF generados por el teclado de sus teléfonos o empleando el reconocimiento del habla.

Los sistemas de IVR difieren por tener *hardware* y *software* diferentes. Encontrar una solución de IVR de un costo efectivo, así como fácil de instalar, mantener y que se pueda integrar fácilmente con otros sistemas (como la recuperación de los datos) generalmente es una prioridad de los usuarios de dicha tecnología.

A modo de resumen las ventajas de un sistema con IVR son:

- Mejor servicio de atención al cliente.
- Reduce las colas de espera y los tiempos de respuesta.
- Reduce el porcentaje de abandono de llamadas.
- Aumenta el número de llamadas atendidas.

- Permite servicio de 24 horas.
- Acceso inmediato y sin esperas.
- Menor costo de atención por llamada.

1.3 Protocolos

Un sistema de IVR eficiente además de reproducir automáticamente mensajes de voz debe brindar la opción de que el usuario que llama pueda hablar directamente con un agente específico del centro debido a que requiere una información adicional o especializada. Para esto, el servidor del sistema IVR debe ser capaz de dirigir la llamada hacia la computadora (cliente) en la cual se encuentra dicho agente. Esto se realiza mediante la interacción de los *softwares* cliente-servidor que a su vez se basa en la tecnología VoIP (Voice Over IP).

VoIP es la transmisión de voz sobre redes IP (*Internet Protocol*). El estándar VoIP define la tecnología que permite encapsular la voz en paquetes para ser transportados sobre redes de datos sin necesidad de disponer de circuitos conmutados, sino que envía múltiples conversaciones a través del mismo canal (circuito virtual) codificadas en paquetes y en flujos independientes.

En VoIP se emplean dos tipos de protocolos, uno para el transporte de la voz en paquetes y otro de señalización. En las redes de computadoras, que se basan en el conjunto de protocolos TCP/IP (*Transmission Control Protocol*), el protocolo de transporte es denominado Protocolo de Transporte en Tiempo Real (Real-time Transport Protocol, RTP).

Existen diversos protocolos de señalización, entre ellos se encuentra H.323, IAX (*Inter-Asterisk eXchange protocol*) y SIP (*Session Initiation Protocol*). Este último es el de mayor uso, y es el que se empleará en este trabajo debido a su extensión y aceptación. SIP es un protocolo desarrollado por el grupo de trabajo IETF (*Internet Engineering Task Force*) empleado para establecer, mantener y finalizar sesiones. Su sintaxis se asemeja mucho a las del protocolo HTTP (*HyperText Transfer Protocol).*

El objetivo de SIP es brindar un estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, la voz y la mensajería instantánea.

1.4 *Software* servidor.

El *software* servidor es el encargado de realizar la mayoría de las funciones en un sistema IVR. Este se encuentra instalado en la computadora a la cual se conecta el *hardware* que funciona de interfaz con la línea telefónica. El mismo debe ser capaz de:

- Comunicarse y controlar el *hardware* CTI para que el servidor pueda atender las llamadas entrantes, reproducir los menús de respuesta de voz interactiva, procesar las entradas de dígitos marcadas por los usuarios y encaminar las llamadas a los distintos agentes conectados al servidor.

- Permitir que el administrador del sistema pueda configurar el registro de usuarios, asignándoles extensiones y garantizando la seguridad en la conexión de estos.

- Posibilitar la edición de los menús IVR, permitiendo personalizar las distintas locuciones que se reproducen al usuario que interactúa con el sistema, así como las acciones que ocurren cuando se marca determinado dígito en el teclado de su aparato telefónico.

1.5 *Software* cliente.

El *software* cliente es el programa que emplean los agentes de la red del centro para comunicarse ya sea entre ellos o con el usuario que realiza la llamada. La función primordial que realiza este *software* es, garantizar la comunicación con el usuario a través del servidor haciendo uso del protocolo SIP anteriormente

mencionado y el transporte de la voz previamente codificada permitiendo el control de los niveles de audio de recepción y transmisión.

CAPITULO 2 . Análisis de *softwares* para sistemas IVR.

Existen varios *softwares* que permiten implementar sistemas IVR. A continuación, se analizan dos de ellos que destacan por ser libres y de código abierto (*FOSS*, Free and Open Source Software), o sea que están liberados bajo licencias que permiten su estudio, modificación y libre uso.

Los escogidos resaltan por su amplio abanico de funcionalidades que compiten con los sistemas propietarios, y por estar en constante desarrollo y mantenimiento, además de contar con una amplia comunidad de usuarios.

2.1 Asterisk.

Asterisk es un programa que permite que una computadora desempeñe funcionalidades de una centralita digital o PBX (*Private Branch Exchange*). Su logotipo se muestra en la figura 2.1. Como cualquier centralita digital, se le puede conectar un número determinado de teléfonos o de *softphone* (*software* cliente) para hacer llamadas entre sí e incluso conectarse a un proveedor de VoIP o bien a una Red Digital de Servicios Integrados (RDSI o ISDN, Integrated Services Digital Network), tanto básicos como primarios. Es un sistema de código abierto, con un amplio número de desarrolladores y seguidores en todo el mundo (Bryant, Madsen, & VanMeggelen, 2013).

Figura 2.1.Logotipo de Asterisk.

Fuente: Google images.

Asterisk está diseñado para correr sobre sistema operativo Linux y actualmente existen distintas distribuciones de ese sistema operativo diseñados

específicamente para implementar centralitas digitales, siendo una de las más empleadas Elastix, desarrollado por la empresa ecuatoriana Palosanto Solutions.

Al mezclar telefonía y servicios de VoIP, Asterisk permite construir arquitecturas de telefonía avanzadas y soluciones de CTI, facilitando la migración gradual de los sistemas existentes en las empresas a las nuevas tecnologías basadas en redes IP.

2.2 FreeSWITCH.

FreeSWITCH fue creado por Anthony Minessale, el cual se desempeñó durante tres años como desarrollador de Asterisk. Durante ese tiempo se encontró con varios problemas en el núcleo de Asterisk que impedían desarrollar sus propias aplicaciones, por lo que se decidió a iniciar esta nueva plataforma con una concepción diferente del núcleo, pero manteniendo el mismo diseño modular que le provee flexibilidad al sistema (Minessale, Collins, Schreiber, & Chandler, 2013). En la figura 2.2 se muestra el logotipo de este programa.

Figura 2.2. Logotipo de FreeSWITCH.
Fuente: Google images.

FreeSWITCH puede funcionar como un *softphone*, un sistema PBX, un *soft-switch*, o como interfaz con otros sistemas PBX *opensource* tales como OpenPBX.org, Bayonne, Yate o Asterisk (Freeswitch, 2017).

FreeSWITCH es liberado bajo la licencia pública de Mozilla (Mozilla Public License, MPL), la cual es una licencia de código abierto que cumple con la definición dada por la *Open Source Initiative* y la *Free Software Foundation*. Permitiendo que el código fuente de FreeSWITCH sea libre para su descarga, revisión, modificación y distribución. Esta licencia también posibilita que las

aplicaciones generadas con el FreeSWITCH puedan ser distribuidas comercialmente, sin necesidad de pagar licencias a los creadores del mismo.

2.3 ¿Por qué escoger FreeSWITCH para un sistema IVR?

Tanto Asterisk como FreeSWITCH permiten la construcción de sistemas telefónicos avanzados como PBX, con diversas aplicaciones como correo de voz, conferencia entre varios usuarios, IVR y aplicaciones de gestión de información. Sin embargo, FreeSWITCH destaca por las siguientes razones:

- Versatilidad: FreeSWITCH provee una gran plataforma para extender sus funcionalidades. Para esto posee varios módulos que permiten que otros programas tomen control sobre él, ya sea a través de una interfaz de red o por lenguajes de *script* como Lua, Perl o JavaScript.

- Los ficheros de configuración de FreeSWITCH son del tipo XML (*eXtensible Markup Language*), un lenguaje de descripción de datos sencillo y popular, lo que hace que sean fáciles de modificar desde un programa para tal fin. No obstante, FreeSWITCH permite además la configuración basada en ficheros de texto plano al igual que Asterisk, por lo que existe compatibilidad entre ambos (Northforge, 2017).

- Es un sistema multiplataforma, siendo posible instalarlo en Linux, IOS X o Windows, al contrario de Asterisk que está diseñado específicamente para Linux.

FreeSWITCH no está exento de inconvenientes, el más notable es la ausencia de una interfaz gráfica amigable para el usuario, que permita modificar fácilmente su configuración. Actualmente FreeSWITCH se visualiza sobre una ventana de consola tal como se muestra en la figura 2.3. En esta se deben introducir los comandos para su manejo, mientras que los ficheros de configuración deben ser modificados manualmente.

Figura 2.3. Ventana del FreeSWITCH.

Elaborado por: el autor.

2.4 Arquitectura de FreeSWITCH.

Una vez analizadas las razones de la elección de FreeSWITCH para realizar un sistema IVR se procede a explicar el funcionamiento del mismo.

La meta del diseño de FreeSWITCH es proveer un sistema modular y escalable alrededor de un núcleo estable, y proveer una robusta interfaz para que los desarrolladores puedan agregar funcionalidades (Minessale, Collins, Schreiber, & Chandler, 2013). Los módulos son cargados según hayan sido configurados el núcleo central, permitiendo seleccionar solo aquellos que convengan a la aplicación en cuestión.

FreeSWITCH tiene muchos tipos diferentes de módulos que se comunican con y a través del núcleo central (Minessale, Collins, Schreiber, & Chandler, 2013). Esta característica es la que hace que este programa sea tan estable y robusto ante posibles fallas, pues si alguno de estos módulos se bloquea o deja de funcionar por alguna razón en particular, el sistema simplemente reinicia esa sección sin interrumpir al resto.

Clasificación de los módulos:

- ***Endpoint***: Módulos de terminación de comunicación, se encuentran aquí agrupados tanto los protocolos de telefonía, como SIP o H.323 como aquellos que permiten el control de la tarjeta de audio de la computadora.

- ***Dialplan***: Tipo de módulo principal, ya que son los que toman las decisiones de encaminamiento de llamadas según los registros previamente configurados, poseen la mayor relevancia dentro del amplio espectro de módulos que conforman al FreeSWITCH.

- ***Application***: Aplicaciones que pueden ser ejecutadas desde el *dialplan*, permitiendo que el sistema pueda realizar diversas acciones como reproducir ficheros de audios, transferir llamadas, pronunciar una frase compuesta de sonidos pregrabados, etc.

- ***Codec***: Codificación de formatos de audio.

- ***Eventhandlers***: Permite a programas externos tomar el control de FreeSWITCH.

- ***Fileformat***: Maneja los formatos de ficheros, como los ficheros de audio WAV.

Algunos de los módulos más importantes:

- Mod_sofia: Es un módulo *Endpoint* que permite a los *softphone* comunicarse entre sí haciendo uso del protocolo de señalización SIP. La base del mismo es la biblioteca Sofia-SIP, un proyecto *opensource* realizado por los laboratorios de Nokia y que facilita la construcción de sistemas de VoIP.

- Mod_event_socket: Es un módulo *Eventhandler*, permite que un programa externo se conecte a FreeSWITCH a través de una interfaz *socket* TCP, permitiendo tomar control total sobre el mismo. Este módulo es el que va a permitir la construcción de una interfaz gráfica de administración.

- Mod_commands: Es un módulo del tipo *Application*, que permite el envío de comandos de configuración y es utilizado por Mod_event_socket.

- Mod_dialplan_xml: Es un módulo de tipo *Dialplan* basado en el formato XML. El plan de llamada o *dialplan* es el corazón de toda PBX. En él se configura el encadenamiento de las llamadas de acuerdo a variados criterios como el número del llamante, la hora del día, la fecha, etc.

- Mod_portaudio: Es un módulo *Endpoint* usado para controlar la tarjeta de audio de la computadora, permitiendo usar FreeSWITCH para originar y recibir llamadas. Está basado en la biblioteca PortAudio, un proyecto *opensource* multiplataforma.

- Mod_say_es: Módulo que permite componer y reproducir frases con la base de datos de sonidos pregrabados en el idioma español.

- Mod_cdr_csv: Este módulo permite la creación de registro de llamadas, almacenando la información de las llamadas realizadas en un fichero de texto.

Cada uno de los módulos se compone de una biblioteca de enlace dinámico (Data Definition Language, DLL) almacenado en el directorio "mod" de la carpeta de FreeSWITCH, así como de ficheros de configuración XML ubicados en la carpeta "conf /autoload_configs". Para la utilización de los mismos es necesario realizar una configuración de sus distintos parámetros.

Para poder transmitir la voz proveniente de la línea telefónica hacia las computadoras conectadas al servidor es necesario realizar la conversión

analógica-digital de esta, para lo cual se emplean los codificadores y decodificadores (*codecs*).

Los *codecs* son algoritmos de compresión de datos de audio para voz, se emplean para disminuir los requerimientos de velocidad de transmisión y carga computacional a costa de variar la calidad de la voz. Algunos *codecs* requieren el pago de licencia a sus propietarios mientras que otros son libres, estos últimos son los que vienen instalados con el FreeSWITCH.

FreeSWITCH tiene integrado en su núcleo el uso por defecto de los *codecs* G.711 con sus dos varientes: ley A y ley μ, el módulo Mod_spandsp permite el uso del G.722 y el GSM (Global System for Mobile communications).

2.4.1 Configuración de FreeSWITCH.

Al ejecutar el FreeSWITCH, se carga en memoria la configuración básica almacenada en el fichero “freeswitch.xml”, que a su vez carga el resto de la configuración que se encuentra fraccionada en varios documentos para lograr una mejor organización. Cada módulo posee su propio fichero de configuración, siendo estos almacenados dentro de la carpeta "conf" en el directorio de FreeSWITCH.

El módulo de mayor importancia resulta ser el *dialplan* o plan de llamadas, que es una lista de acciones que se deben ejecutar cuando se cumplan ciertas condiciones. El *dialplan* se compone de extensiones, que son agrupaciones lógicas de tareas, estas a su vez se componen de condiciones y acciones.

FreeSWITCH constantemente verifica todas las condiciones de las extensiones del *dialplan*, en caso de que se cumpla alguna se procede a ejecutar la secuencia de acciones configurada para dicha extensión.

Las condiciones se construyen con expresiones lógicas en base al conjunto de variables globales predefinidas. Un ejemplo de extensión es el siguiente:

<extensionname="ivr_menu">

<condition field="destination_number" expression="^5000$">

<action application="answer" />

<action application="sleep" data="2000" />

<action application="ivr" data="demo_ivr" />

</condition>

</extension>

En el encabezado de la etiqueta *extension* se define el atributo *name*, que permite diferenciar las extensiones entre sí, luego sigue un bloque conformado por la etiqueta *condition* que conforma la expresión lógica, en este caso la condición es: ¿el campo *destination_number* (número marcado) es igual al valor 5000? En caso de ser verdadera dicha condición se procede a ejecutar en orden secuencial la lista de acciones definidas.

En este ejemplo primero se ejecuta la aplicación *sleep* la cual introduce una demora en milisegundos igual al argumento pasado en el campo *data***,** esto se realiza para asegurar que los recursos de audio se encuentren listos antes de comenzar a reproducir cualquier sonido, la última acción es la ejecución de la aplicación *ivr* la cual transfiere la llamada a un menú de respuesta de voz interactiva llamado "demo_ivr" que debe encontrarse en el fichero de configuración del IVR.

FreeSWITCH permite construir condiciones complejas gracias al uso de las expresiones regulares del lenguaje PERL, las cuales permiten comparar si una secuencia cumple con determinado patrón. Una expresión de este tipo es verdadera si cumple con todas las condiciones impuesta por el patrón y falsa si no cumple con al menos una. Las sintaxis básicas de estas expresiones se muestran en la tabla 2.1.

Tabla 2.1.Sintaxis básica de expresiones PERL.

Patrón	Significado
5000	Se marca un número que contiene 5000.
^5000	Se marca un número que comienza con 5000.
^5000$	Se marca exactamente el número 5000.
\d	Se marca un dígito.
\d\d	Se marcan dos dígitos
\d+	Se marca un grupo cualquiera de dígitos.

Fuente: (Minessale, Collins, Schreiber, & Chandler, 2013)

FreeSWITCH tiene más de 140 aplicaciones en su *dialplan* (Minessale, Collins, Schreiber, & Chandler, 2013). A continuación, se muestran las más importantes y de uso frecuente:

- *answer*: Atiende una solicitud de llamada, abre los recursos de audio para la comunicación.

- *bridge*: Sirve para conectar a dos *endpoints* entre sí, permite redirigir una llamada.

- *playback*: Reproduce un fichero de audio previamente almacenado en el disco duro de la computadora servidor.

- *say*: Esta aplicación permite conformar y reproducir frases, utilizando una base de datos de sonidos pregrabados. Tiene como parámetros el tipo de datos a reproducir, estos pueden ser número, elementos, personas, mensajes, número de teléfono, dirección IP, dirección de correo, etc. El otro argumento es el método de pronunciación, este puede ser de tipo: "pronunciación" (ejemplo: 1023 sería mil-veintitrés), "iteración" (ejemplo: 1023 sería uno-cero-dos-tres) y"cuenta" que es similar a pronunciación, y no aplicable para casos no numéricos.

- *ivr*: Ejecuta un menú de respuesta de voz interactiva, los ficheros de estos se almacenan en la carpeta "conf/ivr_menus".

- *set_global*: Permite crear o cambiar el valor a una variable global.

- *voicemail*: Ejecuta el buzón de voz, este es un menú IVR preelaborado que permite a los usuarios revisar los mensajes que ha recibido.

- *hangup*: Termina una llamada, liberando los recursos de audio asignados.

2.5 *Software* cliente.

El programa cliente es un *softphone*, puede emplearse para comunicarse con otro *softphone* activo en una de las computadoras de una red o con un teléfono de la PSTN a través del servidor. Entre las funciones de estos programas destaca la digitalización de la voz (proceso de muestreo, cuantificación y codificación) proveniente de un micrófono, la formación de paquetes con la voz para su envío a través de una red de datos, así como funcionalidades extras propias de la telefonía como registro de números, función de rediscado, grabación de las llamadas, etcétera.

Actualmente existen varios *softwares* clientes que emplean el protocolo SIP y que son libres, algunos ejemplos son:

- MicroSIP

Programa libre y *opensource*, su interfaz se muestra en la figura 2.4 destaca por su reducido tamaño de menos de 3 *Megabytes*, no obstante, posee funcionalidades de registro de llamadas, lista de contactos y permite configurar de forma sencilla sus parámetros de operación. Su código fuente puede ser descargado desde la página http://www.microsip.org.

Figura 2.4. Pantalla principal de MicroSIP.

Elaborado por: el autor.

- CounterPath X-Lite

Este *software* permite la comunicación tanto de voz como de video, su apariencia simula un teléfono móvil y posee funcionalidades de transferencia de llamada, llamada en espera, respuesta automática y grabación de llamadas. También posee barras de control que manejan los niveles de audio del micrófono y de la salida de audio de la computadora.

La ventana principal de mismo se muestra en la figura 2.5. Este programa es de uso gratuito y presenta como ventaja con respecto a otros que permite asignarle a la comunicación parámetros de Calidad de Servicio (Quality of Service, QoS) priorizando los paquetes, garantizando que en redes congestionadas se disminuya la posible latencia que pueda ocurrir. También brinda la posibilidad de modificar los parámetros de los códec de audios empleados.

La elección del *softphone* a emplear es propia del administrador del sistema dado que cualquiera de ellos cumple con las funciones básicas de este tipo de *software*. Para conectarse al servidor se debe configurar la dirección IP del servidor (computadora donde se ejecuta el FreeSWITCH) y luego registrarse con una credencial previamente insertada en el *dialplan*.

Figura 2.5. Pantalla principal de CounterPath X-Lite.

Elaborado por: el autor.

CAPITULO 3 . Diseño de un sistema IVR de perfil amplio.

A continuación, se presenta el esquema en bloques del sistema IVR propuesto. El mismo se concibió teniendo en cuenta que debe ser un entorno de perfil amplio fácilmente configurable para la gestión de información en hospitales, bancos y otros centros de interés.

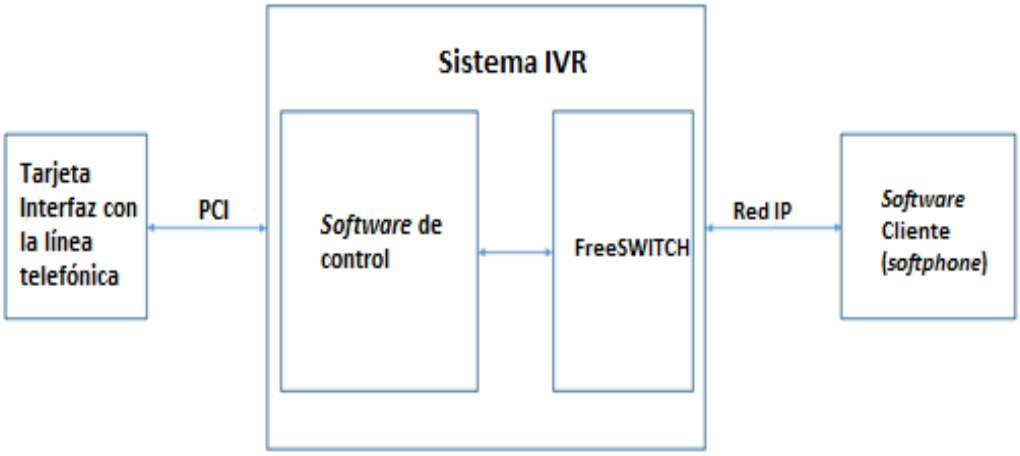


Figura 3.1.Esquema en bloques del sistema IVR.

Elaborador por: el autor.

A partir de este esquema se procede a explicar cada uno de los bloques que lo componen.

3.1 Tarjeta de interfaz con la línea telefónica.

La tarjeta de interfaz con la línea telefónica se encuentra conectada a la computadora servidor mediante el puerto PCI y realiza las siguientes funciones:

- Garantiza el colgado y descolgado de la línea telefónica.
- Analiza los tonos de progreso de llamada.
- Detecta la señal de timbre.
- Detecta los tonos DTMF generados por el llamante.
- Permite la comunicación *full duplex*.

Entre los principales fabricantes de estas tarjetas se encuentra la compañía Digium, creadora del *software* Asterisk (véase figura 3.2), y la compañía Sangoma.

Figura 3.2. Tarjetas de interfaz con la línea telefónica de Digium.
Fuente: **(Digium, 2015)**

Para el diseño del presente sistema se seleccionan las tarjetas analógicas de la compañía Sangoma principalmente por su compatibilidad con el *software* FreeSWITCH y por tener las siguientes facilidades técnicas:

- Compatible con todas las placas base (*motherboards* en inglés) comercialmente disponibles.
- Compatible con dispositivos PCI/PCI (Peripheral Component Interconnect) Express. No presenta ningún problema de compatibilidad con IRQ (*interrupt request*).
- Módulos de cancelación de eco y mejoramiento de voz.
- Bajo consumo de los recursos de la CPU (Central Processing Unit).
- *Firmware* fácilmente actualizable.

En la serie de tarjetas analógicas de Sangoma se encuentran los modelos A200, A400 y B600.

En la tabla 3.1 se aprecia una comparación de los modelos anteriormente citados.

Tabla 3.1. Serie de tarjetas analógicas de Sangoma.

Serie análogas			
	A200	A400	B600
Número de puertos	2-24 (Expandible)	2-24 (Expandible)	4
Interfaces	FXO/FXS Modular	FXO/FXS Modular	4 FXO y 1 FXS
Expansion de arquitectura	Hasta 2 módulos por tarjeta Hasta 5 placas hija	Hasta 6 módulos por tarjeta Hasta 2 placas hija	-
Velocidad de transferencia	132MB/sec	132MB/sec	132MB/sec
Garantía	De por vida	De por vida	5 años

Fuente: (Sangoma, 2016)

Para una calidad de voz, óptima cancelación de eco y 24 puertos de comunicación se debe elegir los modelos A200 y A400. Para sistemas pequeños y estables que necesiten una reducción de costo y seguir disfrutando del audio Premium de Sangoma se debe elegir la tarjeta de voz B600 diseñada particularmente para instalaciones de bajo presupuesto. (véase figura 3.3)

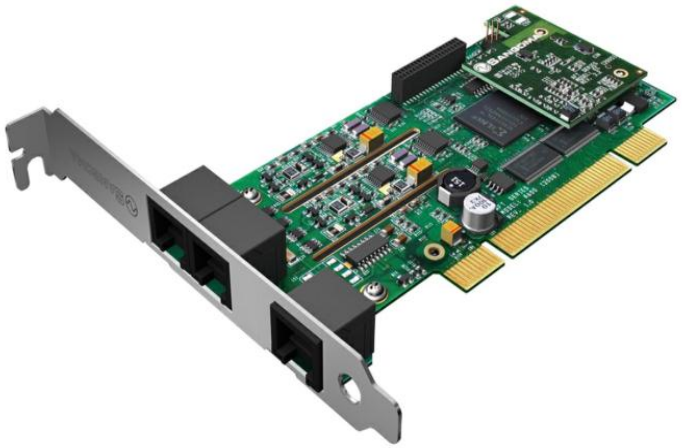


Figura 3.3.Tarjeta B600 de la compañía Sangoma.
Fuente: (Sangoma, 2016)

3.1.1 Especificaciones técnicas:

- Compatible con Asterisk, FreeSWITCH y otro PBX de código abierto o patentado, Switch, IVR o aplicaciones de portales VoIP.
- Interfaz PCI sincrónico para todos los puertos FXO/FXS.
- Interfaz PCI a 132 Mbytes/s para intervención mínima del procesador del sistema anfitrión (host).
- Compatibilidad con autodetección de 5V y buses PCI 3.3V
- PCI compatible con todas las placas madres disponibles en el mercado, y compartición de interrupciones propias de PCI
- Hardware inteligente: Matriz de Entradas Programables por Campo descargable con múltiples modos operacionales, ampliable para que las nuevas funciones puedan ser agregadas cuando se encuentren disponibles.
- Margen de temperatura: 0 – 50 °C
- *Hardware* DSP Opcional con placa de cancelación de eco:
  - *Hardware* con cancelación de eco G.168–2002
  - Decodificación y reconocimiento de tono DTMF
  - Mejora en la calidad de voz: protección musical, control de eco acústico y reducción de ruido adaptativo
- El tamaño físico de la tarjeta no se incrementa y no se requiere ranura adicional.
- Sistemas Operativos compatibles:
  - Linux (todas las versiones, lanzamientos y distribuciones desde el 1.0 en adelante)
- Windows XP o superior (véase figura 3.4)

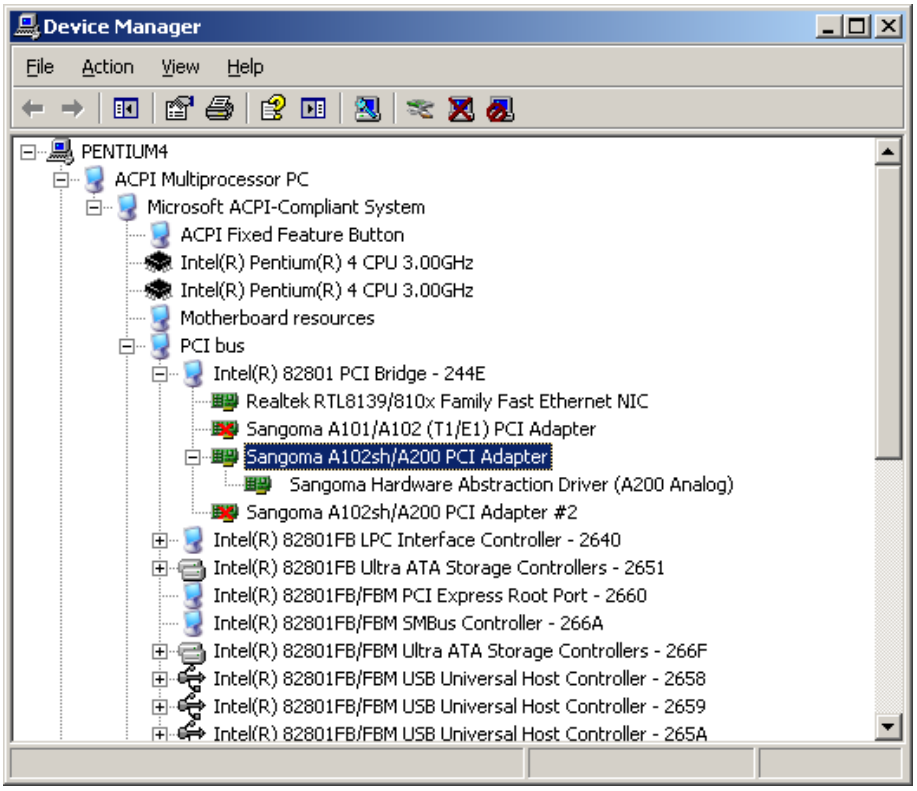


Figura 3.4. Driver de tarjeta Sangoma A200 vista desde el administrador de dispositivos de Windows.

Elaborado por: El autor.

3.1.2 Tarjetas Sangoma y el FreeSWITCH.

FreeSWITCH es completamente compatible con muchos tipos de tarjetas de interfaz con la línea telefónica. Varios de esos tipos de tarjetas son fabricadas por Sangoma, Digium, OpenVox, Rhino Technologies, RedFone entre otros.

Los desarrolladores de FreeSWITCH originalmente crearon una biblioteca llamada OpenZAP. Esta capa de abstracción permitía a FreeSWITCH comunicarse con las tarjetas de Digium y Sangoma. FreeTDM, patrocinado por *Sangoma Technologies Corporation*, es una nueva capa de abstracción que ha reemplazado completamente a OpenZAP.

Sangoma ha concentrado todos sus esfuerzos en crear las mejores APIs (*Application Programming Interface*), *drivers* y *hardware* del mercado. Ha adoptado la plataforma FreeTDM como el estándar por defecto para la señalación

basada en multiplex por división en tiempo (Time Division Multiplexing, TDM) y la API multimedia (véase figura 3.5). Actualmente usa la capa API de FreeTDM como una API unificada de señalización y multimedia para los sistemas operativos Linux y Windows para clientes que estén buscado desarrollar aplicaciones de telefonía sobre *hardware* de Sangoma.

La compañía se mantiene como desarrollador *open source* del proyecto FreeTDM y también colabora de forma activa en el desarrollo de FreeSWITCH.

En resumen, FreeTDM es una biblioteca que permite al FreeSWITCH comunicarse con las tarjetas de interfaz con la línea de fabricantes como Sangoma y Digium. Implementa un alto nivel de unificación entre los módulos de señalización y los módulos multimedia o I/O (entrada/salida). Hay tres tipos de módulos I/O en dependencia del *hardware* a utilizar.

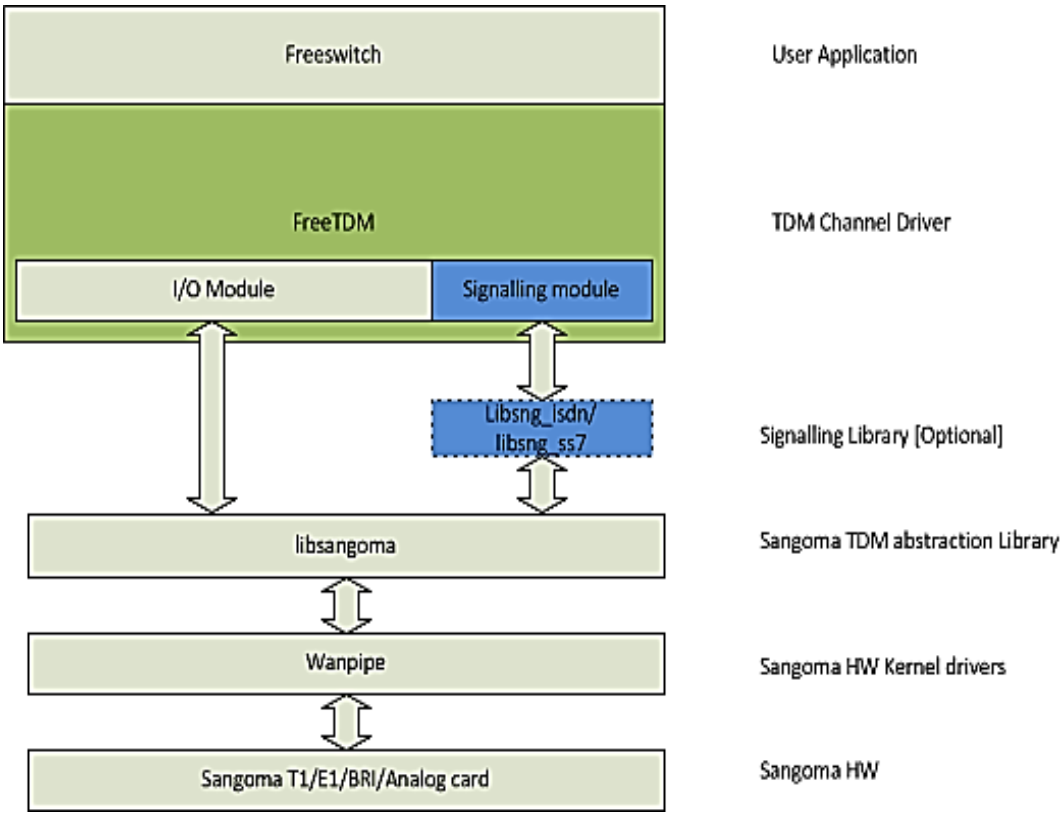


Figura 3.5. FreeTDM.

Fuente: (Freeswitch.org, 2017).

En el presente proyecto se usa *hardware* de la compañía Sangoma por lo que el módulo I/O a emplear es el ftmod_wanpipe. Este módulo sólo puede ser usado para *hardware* Sangoma pues usa la API nativa de dichas tarjetas (libsangoma) la cual se instala junto con los drivers de la tarjeta en cuestión.

Los módulos I/O son los encargados de cuidar la escritura y lectura de datos y de enviar comandos a ejecutar por la tarjeta. Quien se encarga de intercambiar señalización con la línea telefónica es el *Signalling module*.

FreeTDM soporta señalizaciones como PRI, BRI, SS7, MFC-R2, analógicas entre otras. De acuerdo al tipo de señalización debe elegirse la configuración de uno de los siguientes módulos:

- ISDN Modules
- SS7 Modules
- Analog
- MFC-R2

Para la señalización analógica debe emplearse el módulo *Analog* (**ftmod_analog**). Este módulo ofrece soporte FXO/FXS genérico para diversas tarjetas de la compañía Sangoma.

3.2 Servidor del sistema IVR.

Se desarrolló un *software* servidor conformado por un *software* de control y el FreeSWITCH como motor de telefonía.

El uso del FreeSWITCH dentro de la aplicación permitirá el manejo de la tarjeta de interfaz con la línea telefónica, la reproducción de los menús IVR y el registro de usuarios. Debido a que este programa carece de una interfaz gráfica de configuración se implementó un *software* de control que realiza las siguientes funciones:

- **Control de la tarjeta de interfaz con la línea telefónica:** Realiza el control de la tarjeta para iniciar una comunicación, aceptar una llamada entrante, conocer el estado de la línea telefónica, identificar los tonos DTMF marcados y realizar la marcación de dígitos telefónicos. Para esta función se usa la biblioteca FreeTDM y dentro de ella los módulos *ftmod_analog* y *ftmod_wanpipe*.

- **Control de FreeSWITCH:** Como el funcionamiento de FreeSWITCH se basa en la introducción de comandos de control y se necesita que este trabaje de forma automática sin intervención de un operador humano en cada momento, el *software* elaborado se encarga de realizar esta operación. Para lograr lo anterior se comunica con el FreeSWITCH a través de un *socket* (interfaz de programación para redes TCP) haciendo uso del módulo Mod_event_socket. Esta función es la que garantiza que el sistema trabaje de forma autónoma, dado que es el *software* de control es quien se encarga de enviar los comandos para recibir y encaminar las llamadas o registrar a los usuarios.

- **Configuración de ficheros XML de FreeSWITCH:** Dado que FreeSWITCH trabaja a partir de sus ficheros de configuración, es imprescindible modificar los mismos para lograr las funcionalidades del sistema IVR que se desee en cada aplicación en cuestión. Esta es una función bastante compleja, pues la configuración de FreeSWITCH incluye: registro de usuarios, *dialplan* y la configuración de los diversos módulos que lo componen. Para esto, el *software* de control implementado, carga, modifica y salva estos ficheros de forma transparente para el administrador del sistema.

- **Creación de menús IVR:** El *software* de control permite la creación de menús de respuesta de voz interactiva, permitiendo editar las posibles entradas de marcación así como las distintas locuciones que se reproducen en cada momento. Esta funcionalidad es clave dentro del

sistema porque permite personalizar y adaptar la aplicación de acuerdo al lugar donde se implemente.

Para la elaboración del *software* de control se empleó el lenguaje C#, creado y estandarizado por la empresa Microsoft, el cual destaca por ser orientado a objetos, permitiendo programar *software* de alta calidad, reutilizable y eficiente. Este lenguaje puede ser empleado para desarrollar programas para distintos sistemas operativos y arquitecturas de máquina, ya que estos son compilados en tiempo de ejecución por el intérprete, ya sea el .Net en Windows o por la plataforma Mono en Linux.

El entorno de desarrollo integrado (Integrated Development Environment, IDE) empleado para programar el *software* es el Visual Studio (véase figura 3.6), en su versión *Community Edition* 2015. Este entorno profesional no requiere pago de licencia por lo que puede ser usado por empresas pequeñas, en trabajos *open source* o en proyectos universitarios.

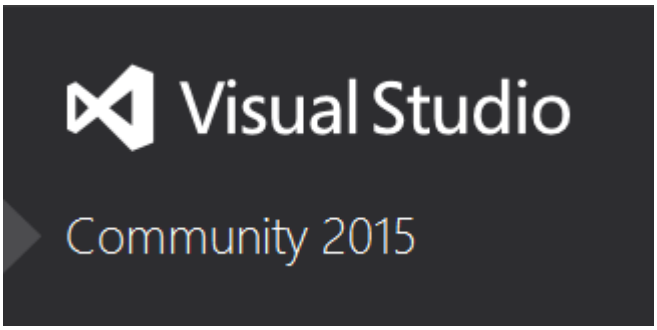


Figura 3.6. Logotipo de Visual Studio.
Fuente: Google images.

El *software* de control se dividió en varias partes tal como se muestra en la figura 3.7, siguiendo un principio de la programación orientada a objetos en el cual se dividen lo más posible todas las estructuras que componen el sistema. Esto permite la reutilización del código, pues al existir bloques con funcionalidades específicas se pueden emplear en más de una ocasión. Otro beneficio de fraccionar el código es que resulta más sencillo de mantener, corregir y documentar.

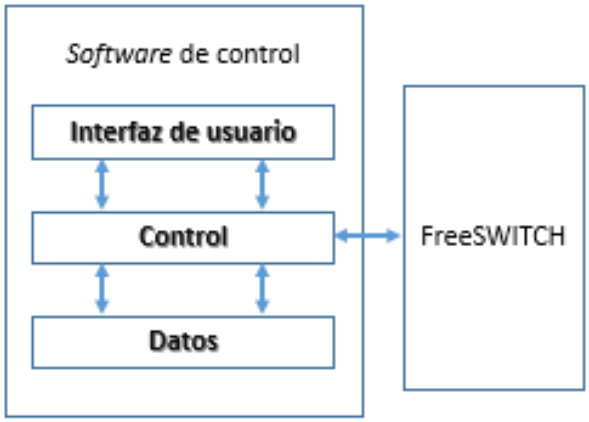


Figura 3.7. Partes que componen el software de control.

Elaborado por: El autor.

La primera parte que se elaboró es el nivel de datos, en este se crearon todas las equivalencias de los ficheros de configuración XML (Extensible Markup Language) del FreeSWITCH a su correspondiente representación como un objeto dentro del lenguaje C#. El beneficio que se obtiene al trabajar los archivos XML como un objeto es un mayor control sobre los mismos. Este nivel es independiente de la arquitectura sobre la que se ejecuta el *software*, permitiendo la portabilidad del código a otras plataformas.

El siguiente nivel es el de control, este se encarga de proveer los mecanismos que permiten convertir los ficheros XML a su correspondiente del nivel uno y viceversa. En este nivel se implementaron todas las funciones que permiten crear, modificar y guardar los archivos del directorio de configuración del FreeSWITCH, así como el control de las extensiones del *dialplan*, elemento indispensable para que el sistema funcione de forma adecuada.

La última parte es el de la interfaz de usuario del programa, este hace uso de los anteriores niveles para brindar de esta forma un *software* sencillo de emplear gracias

a que abstrae la complejidad del FreeSWITCH. En este nivel es donde se llevan a cabo todas las tareas y algoritmos planteados anteriormente. Los dos primeros niveles se desempeñan como bibliotecas de *software* o DLL, y son enlazadas por el tercer nivel que es un ejecutable.

A parte de la realización del *software* de control fue necesario configurar los módulos del FreeSWITCH empleados para realizar el servidor IVR. Estos fueron el Mod_Sofia y el Mod_DialPlan. El primero implementa el conjunto de protocolos SIP necesarios para la interconexión con los usuarios, para su correcto funcionamiento es necesario establecer la dirección IP de la máquina donde se conecta la tarjeta, esto lo realiza el *software* de control de forma automática.

El plan de llamadas se realizó en base a las funciones que se brindan cuando se diseñan los menús IVR. Para esto se hizo uso de las aplicaciones *bridge*, *answer*, *ivr*, *playback*, *voicemail* y otros. El plan de llamadas se muestra de forma íntegra en el Anexo 1.

3.3 Condiciones iniciales y configuración.

Para el correcto funcionamiento del servidor es imprescindible que el FreeSWITCH se encuentre en ejecución, pues es el que permite el control de la tarjeta de interfaz con la línea, la interconexión de los usuarios en red, así como las funcionalidades del menú IVR.

Para poder establecer y responder llamadas desde el exterior a través de la línea telefónica es necesario que la tarjeta de interfaz con la línea se encuentre conectada y exista comunicación con la misma.

Para vincular el FreeSWITCH con la tarjeta de interfaz con la línea telefónica de Sangoma debe configurarse primeramente la biblioteca FreeTDM a través del archivo de texto freetdm.conf que está ubicado por defecto en la dirección de instalación del FreeSWITCH.

Esta configuración por defecto se modifica en dependencia del *hardware* a utilizar siguiendo ciertas reglas de sintaxis que pueden consultarse en la página web de FreeSWITCH (Boteler, 2016).

Un breve fragmento de código de cómo quedaría la configuración para el diseño propuesto sería:

```
[span wanpipe myWanpipeSpan]; módulo wanpipe para tarjetas Sangoma
trunk_type=> FXO
txgain=> 3.5; ganancia del audio de transmisión
rxgain=> 3.5; ganancia del audio de recepción
fxo-channel => 1:1
```

Luego se procede a configurar el módulo I/O a utilizar que, en este caso, sería Wanpipe, que como se explicó anteriormente, se emplea para trabajar con tarjetas Sangoma. Esta configuración se realiza modificando el archivo de texto wanpipe.conf.

Una vez que estén configurados los ficheros anteriores se procede a configurar el fichero freetdm.conf.xml que se encuentra en la dirección:

C:\Program Files \FreeSWITCH\conf\autoload_configs (Esta dirección puede variar en dependencia del sistema operativo)

En esta configuración se le define al FreeSWITCH la configuración para cada tipo de señalización y a donde debe enviar las llamadas entrantes. A continuación, se aprecia un breve fragmento de código de dicha configuración:

```
<analog_spans>
     <span name=" myWanpipeSpan ">
       <!--<param name="hold-music" value="$${moh_uri}"/>-->
       <param name="dialplan" value="XML"/>
       <param name="context" value="Incoming-FXO"/>
       <param name="tonegroup" value="fr"/>
       <param name="enable_callerid" value="true"/>
       <param name="enable-analog-option" value="call-swap"/>
       <!--<param name="enable-analog-option" value="3-way"/>-->
     </span>
```

El *software* de control diseñado, al iniciarse, carga su propio fichero de configuración llamado "conf.xml", donde se encuentra la dirección del FreeSWITCH para luego proceder a su ejecución realizando de forma automática todas las configuraciones antes mostradas, cargando los ficheros XML del FreeSWITCH que permiten poner a trabajar el mismo como PBX, estos son los ficheros del directorio de usuarios, ubicados dentro de la carpeta "conf/directory"; los ficheros del *dialplan*, que se encuentran dentro de "conf/dialplan", los menús IVR, ubicados en "conf/ivr_menus" así como los registros de llamadas almacenados en la carpeta "log".

El FreeSWITCH se configuró para que pueda ser controlado externamente, gracias al módulo Mod_event_socket. El *software* de control se diseñó para que al iniciarse se conecte con él. Esta comunicación es la que permite que el administrador del sistema no tenga que interactuar con el FreeSWITCH, dado que es el programa el que realiza el envío de comandos.

Los comandos empleados en esta comunicación son:

- ***auth*:** Permite establecer la comunicación al realizar la autentificación del *software* con el FreeSWITCH, se pasa como parámetro una contraseña previamente almacenada dentro de la configuración del módulo Mod_event_socket.

- ***status*:** Con este comando el *software* de control indica al FreeSWITCH que debe devolver un conjunto de datos que informan sobre su estado de ejecución, como el tiempo que lleva en funcionamiento, la versión, si existe solicitud de llamada por parte de algún usuario de la red, etcétera.

- ***reloadxml*:** El envío de este comando provoca que el FreeSWITCH cargue nuevamente su configuración, esta instrucción debe ser enviada cada vez que se modifique cualquier aspecto de funcionamiento del servidor, como cuando se agrega o elimina un usuario o cuando se modifica un menú IVR, para que este tenga efecto inmediato.

- ***global_getvar, global_setvar*:** Estos dos comandos se emplean cuando el *software* de control revisa o modifica alguna de las variables globales de configuración del FreeSWITCH.

Constantemente el *software* verifica el estado de la conexión con el FreeSWITCH enviando el comando *status*, para mostrar la información devuelta por el mismo en la interfaz gráfica diseñada. La información devuelta incluye la versión del FreeSWITCH, el tiempo de funcionamiento y el estado de los usuarios de la red.

Se cuenta con una base de datos (Data Base, DB) externa MySQL. Se puede realizar consultas de manera directa desde la API Freeswitch a través del módulo mod_odbc_query cargado. Para poder utilizar el modulo, se requiere haber configurado el conector Unix ODBC de acuerdo al sistema operativo, una vez hecho eso, se configura el archivo odbc_query.conf.xml en autoload_configs.

```
<configuration name="odbc_query.conf" description="ODBC Query Module">
<settings>
<param name="odbc-dsn" value="freeswitch:freeswitch:secret"/>
</settings>
</configuration>
```

Luego de realizar los pasos anteriores e iniciar Freeswitch, se coloca el siguiente comando para consultar la DB externa ya establecida, donde deben retornar los datos:

```
odbc_query [txt|tab|xml] SELECT * FROM table WHERE id=x
```

3.4 Descripción de la interfaz de usuario.

El *software* de control se diseñó para que el sistema tenga una interfaz amigable, sencilla e intuitiva de usar, brindando solo las opciones necesarias para la configuración básica del sistema abstrayendo la complejidad de la configuración

del FreeSWITCH. Esta interfaz permite que el sistema sea fácilmente configurable para ser usado en cualquier centro de información.

3.4.1 Ventana principal.

La ventana principal se compone de tres pestañas, siendo la primera la que se muestra en la figura 3.8.

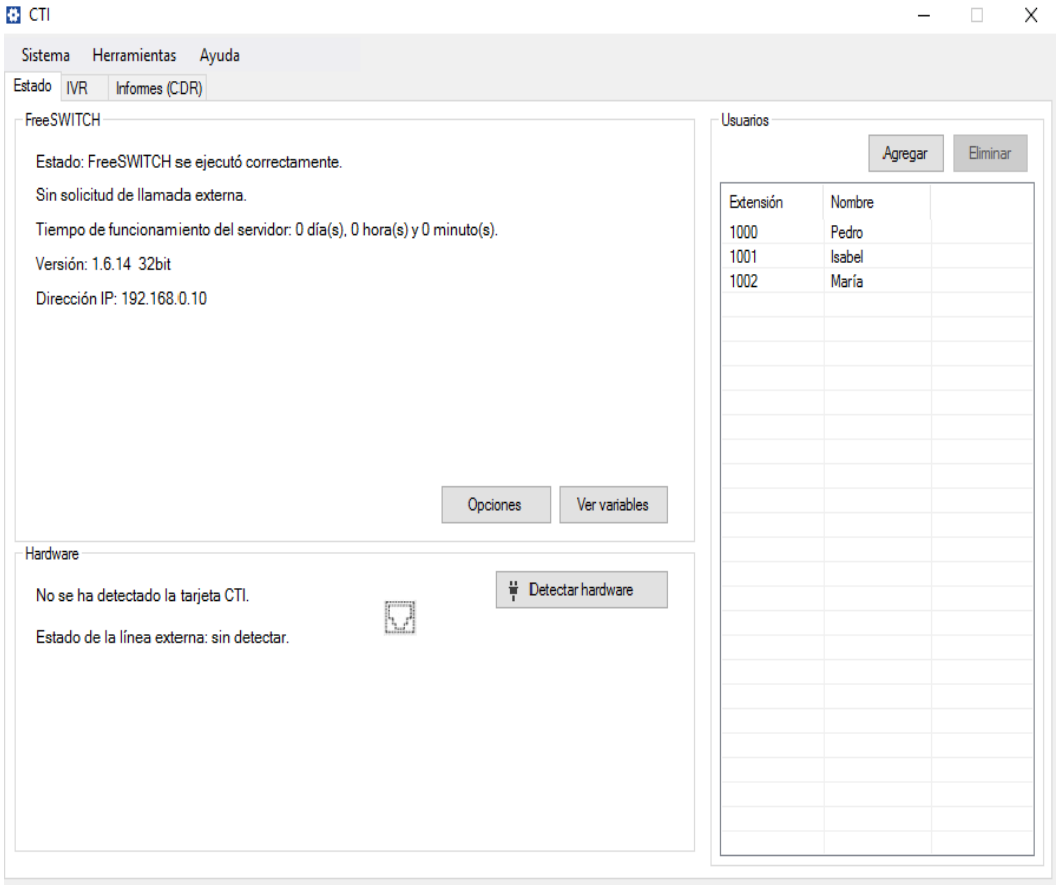


Figura 3.8. Ventana principal de la aplicación.
Elaborado por: El autor.

La primera pestaña se nombra Estado, esta se divide en tres secciones a la vez, una donde se muestra información relacionada con el FreeSWITCH, otra donde se informa sobre el estado de conexión de la tarjeta de interfaz con la línea telefónica y una tercera donde se permite la configuración de los usuarios en red del sistema.

Los datos que se muestran del FreeSWITCH son su estado de ejecución, el tiempo de funcionamiento del servidor, la versión instalada y la dirección IP del servidor. En esta sección también se encuentran accesos a las opciones de configuración del *software* de control.

Al dar clic sobre el botón "Opciones" se abre la ventana que se muestra en la figura 3.9, la cual permite modificar la dirección donde se encuentra instalado el FreeSWITCH, así como la posibilidad de reiniciar la configuración del mismo a sus valores iniciales, en caso de que este no trabaje de la forma adecuada.

Al reiniciar la configuración del FreeSWITCH se eliminan los usuarios registrados, los menús IVR creados y el registro de llamadas.

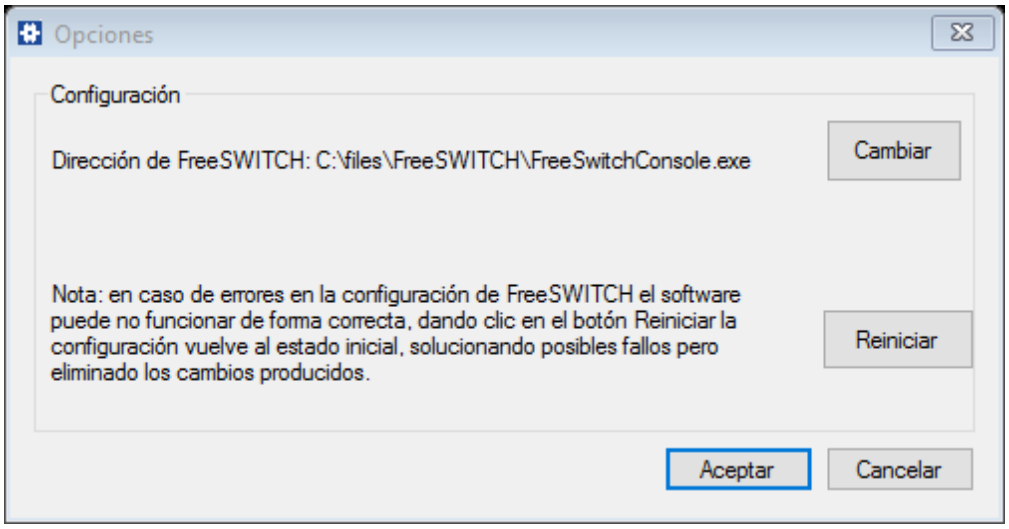


Figura 3.9. Ventana de opciones.

Elaborado por: el autor.

En la segunda sección de la pestaña de Estado, se informa sobre la conexión de la tarjeta de interfaz con la línea y el estado de la línea telefónica (colgado, en uso, sin identificar), así como los dígitos DTMF marcados por los usuarios que interactúan con el servidor. Al dar clic en el botón "Detectar *hardware"* se ejecuta el algoritmo programado para identificar y configurar de forma automática la tarjeta de interfaz con la línea telefónica.

En esta sección se brinda la posibilidad al administrador de poder realizar llamadas desde la máquina servidor, a través de una interfaz telefónica programada en el *software* de control que realiza las funciones básicas de un teléfono convencional, estas son el colgado, descolgado y marcación de dígitos.

Para que el administrador pueda realizar llamadas desde el servidor debe emplear auriculares con micrófono conectados a la tarjeta de audio de la computadora. Al dar clic en la opción "Emplear interfaz telefónica para realizar llamadas" se activa este modo de trabajo.

En la tercera sección de la primera pestaña se diseñaron los mecanismos que permiten al administrador del sistema configurar los parámetros de los usuarios conectados en red al sistema. Dichos usuarios son los posibles agentes de la empresa encargados de atender la llamada en caso de que el llamante necesite información adicional que el sistema IVR no brinda automáticamente.

Para que un usuario tenga acceso a las funcionalidades del sistema basta con que quede registrado su número o extensión, para esto con dar clic en el botón "Agregar" o en el menú Herramientas > Usuarios > Agregar, se abre la ventana de Agregar usuario, mostrada en la figura 3.10, donde se deben introducir las opciones básicas necesarias para agregar un nuevo usuario. Estas son:

Nombre: Nombre con que se presenta cuando realiza una llamada a otro usuario.
Extensión: Número telefónico de la extensión.
Contraseña: Contraseña para registrarse.
Contraseña de correo de voz: Contraseña para acceder al correo de voz.

Una vez completados todos los datos, el *software* verifica que sean correctos y que la extensión no esté en uso, de ser así crea el fichero de configuración XML del usuario en la carpeta de configuración del FreeSWITCH, incluye la extensión del mismo en el *dialplan* y por último envía el comando de reinicio que provoca

que FreeSWITCH cargue la nueva configuración, quedando registrado ya el nuevo usuario.

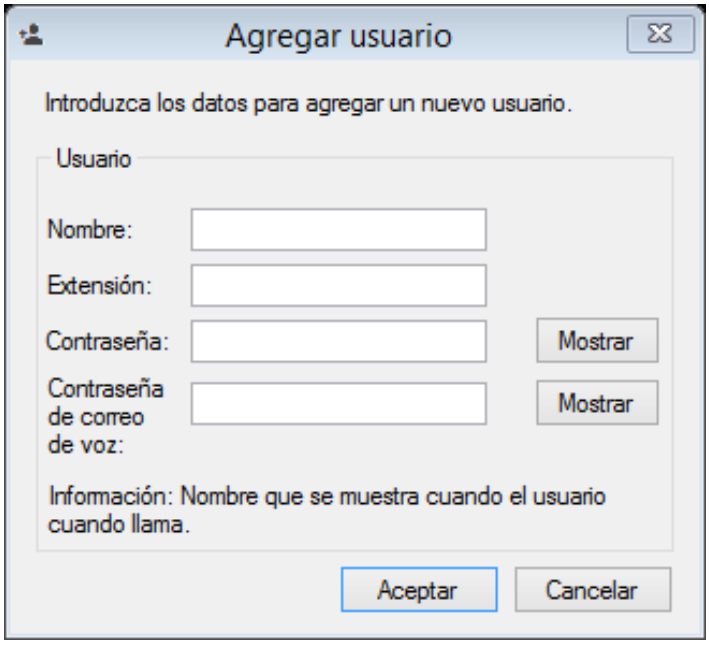


Figura 3.10. Ventana de agregar usuario.

Elaborado por: El autor.

Para eliminar un usuario basta con seleccionarlo en el listado y dar clic en el botón Eliminar o en el menú Herramientas > Usuarios > Eliminar, el *software* elimina entonces el fichero XML correspondiente a ese usuario de la carpeta de configuración, elimina la extensión del mismo del *dialplan* y envía el comando de reinicio al FreeSWITCH que provoca que este cargue nuevamente la configuración, quedando eliminado dicho usuario.

Como se mencionó anteriormente, es el *software* de control quien realiza la configuración del FreeSWITCH de forma transparente para el administrador del sistema, pues para esto simplemente agregó los datos de un usuario al dar clic en un botón, pero en realidad para que el servidor permita conexiones es necesario realizar toda una serie de acciones, que de realizarlas de forma manual implicarían mucho esfuerzo y tiempo.

3.5 Algoritmos de atención de llamadas y menú IVR.

Para acceder al sistema IVR el usuario debe marcar el número telefónico del centro en cuestión. Una vez hecho esto la llamada será tomada y el usuario escuchará la locución del menú principal. Dicha locución le invita a marcar dígitos en el teclado para acceder a cada una de las opciones. En la figura 3.11 se muestra un ejemplo del sistema IVR aplicado a una empresa de seguros.

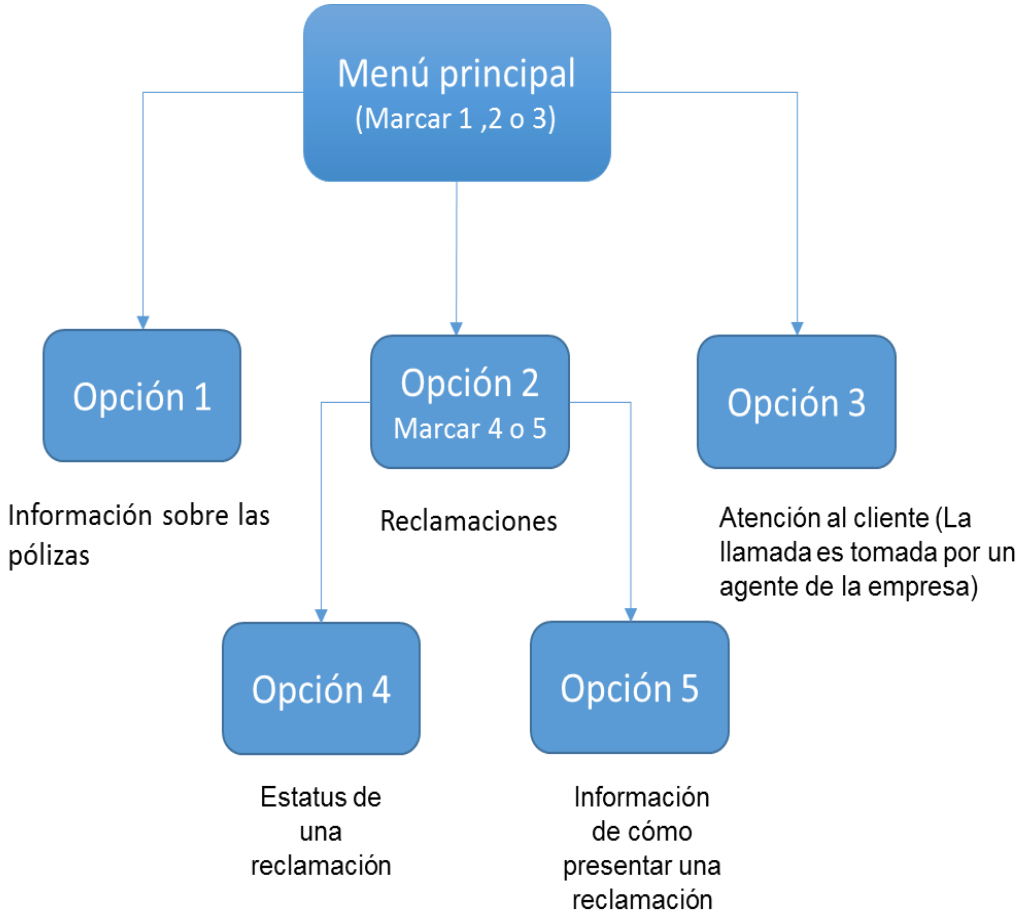


Figura 3.11. Menú IVR.

Elaborado por: el autor.

El sistema puede configurarse con cuantas opciones se desee, para esto el administrador debe grabar previamente los mensajes de respuesta automáticos para cada caso. Luego que el usuario escuche la información deseada el sistema le envía nuevamente al menú principal.

3.6 Comprobación del funcionamiento de la propuesta.

En la figura 3.12 se aprecia una configuración de ejemplo del menú IVR en el *software* de control diseñado. Se establecieron 3 opciones para el menú IVR. Si el usuario presiona el número 3 se transfiere la llamada a la extensión 5002, si presiona el número 2 se transfiere la llamada a la extensión 1030 y por último si presiona el número 1 se reproduce un sonido pregrabado. Esta configuración sería para el menú IVR inicial pero pueden configurarse tantos menús secundarios como se desee mediente el botón agregar. También se configuran los distintos sonidos pregrabados para cada caso.

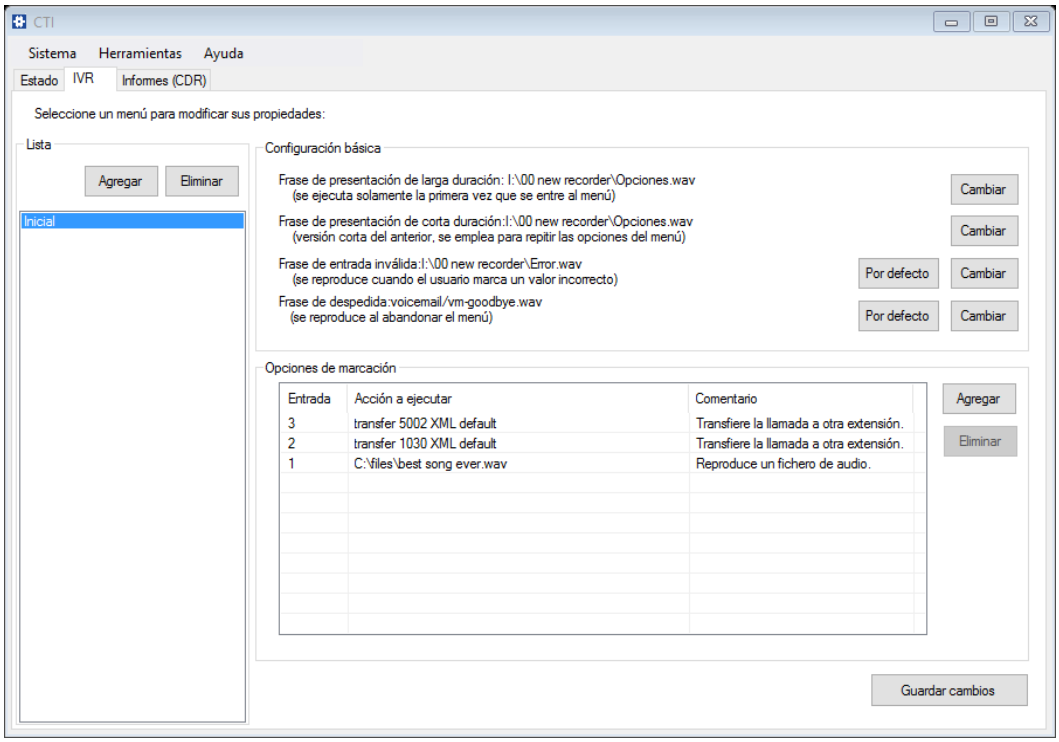


Figura 3.12. Configuración del menú IVR.

Elaborado por: el autor.

En la figura 3.13 se muestra distintos usuarios agregados al sistema, por ejemplo, estos serían los agentes de la empresa dedicados a suministrar atención al cliente.

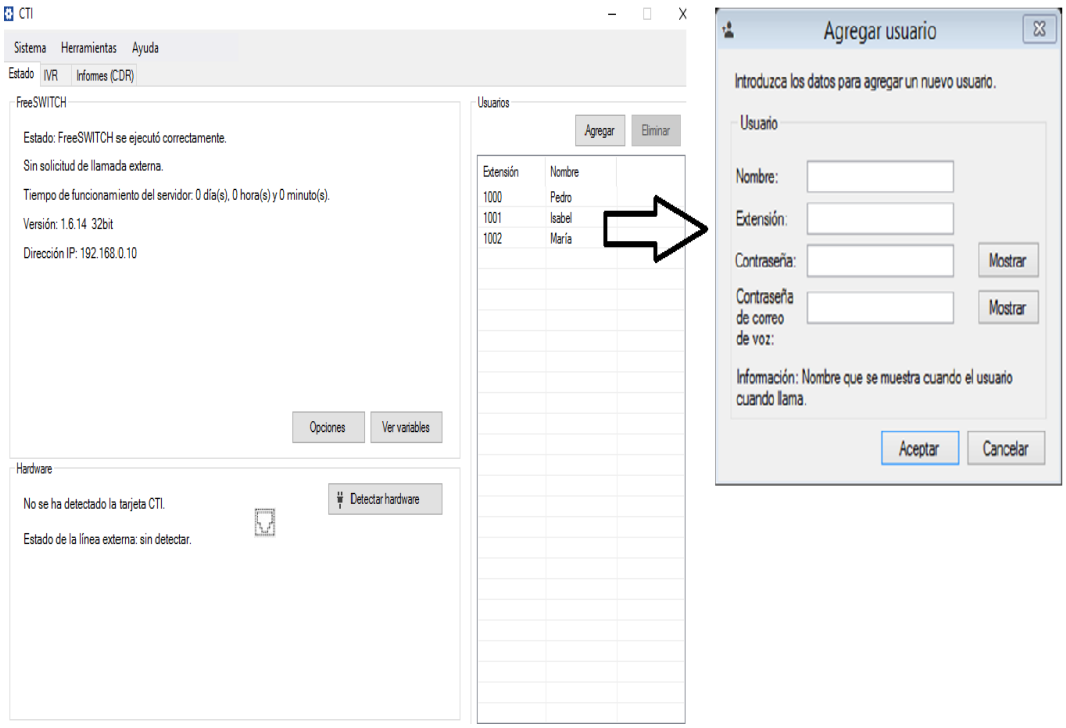


Figura 3.13.Usuarios registros en la red.

Elaborado por: el autor.

Por último, en la figura 3.14 se aprecia el registro de llamadas efectuadas al sistema, todas cursadas con efectividad.

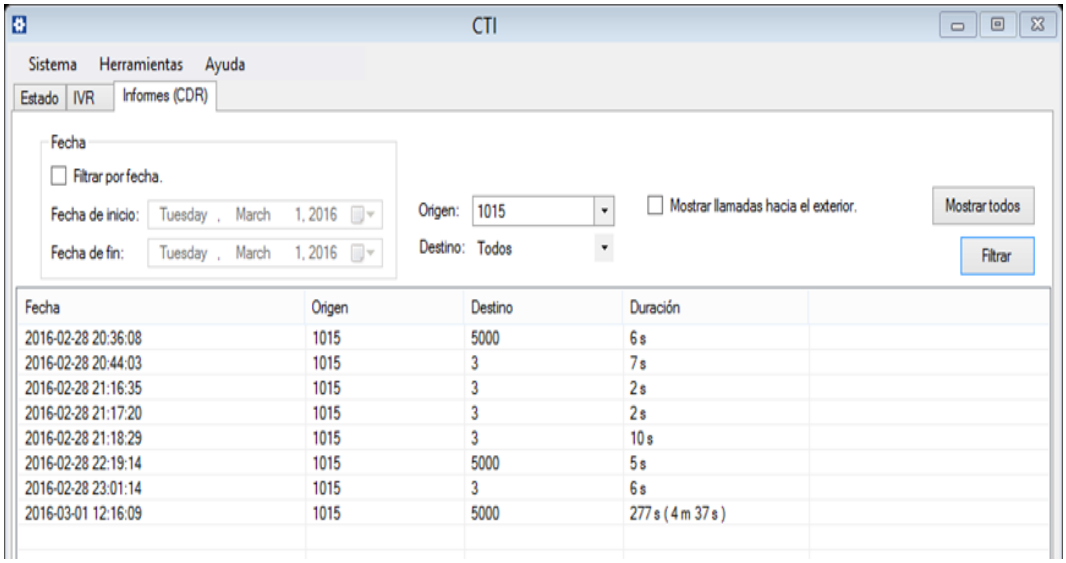


Figura 3.14. Registro de llamadas.

Elaborado por: el autor.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- Del análisis de las características de los softwares Asterisk y FreeSWITCH realizado se concluye que el FreeSWITCH es la mejor alternativa para implementar el sistema IVR propuesto.

- Se comprobó que es posible configurar al FreeSWITCH a través de un entorno de alto nivel para brindar más funcionalidades al sistema IVR.

- Se desarrolló un *software* servidor que posee una interfaz amigable que propicia:
  - ✓ Modificación del registro de usuarios.
  - ✓ Creación de menús IVR personalizables.
  - ✓ Acceso a informes del registro de llamadas.
  - ✓ Control de la tarjeta de interfaz con la línea.

- El sistema realizado cumple con el objetivo de ser de perfil amplio, lo que permite ser fácilmente configurado para emplearse en una gran diversidad de centros que requieran automatizar su información pública.

- Se comprobó el correcto funcionamiento del *software* mediante la recepción de llamadas de pruebas. Las mismas fueron atendidas por el menú IVR o transferidas a usuarios de la red de computadoras. Las pruebas realizadas permitieron constatar el correcto funcionamiento tanto del *software* servidor como del cliente.

Recomendaciones

1. Agregar más opciones al *software* de control que permitan ampliar las prestaciones del sistema.
2. Realizar la implementación práctica del sistema en un centro específico.

REFERENCIAS BIBLIOGRÁFICAS

Boteler, J. (2016). *FreeTDM.* Obtenido de https://freeswitch.org: https://freeswitch.org/confluence/display/FREESWITCH/FreeTDM

Bryant, R., Madsen, L., & VanMeggelen, J. (2013). *Asterisk: The Definitive Guide, Fourth Edition.* Sebastopol, CA 95472: O'Reilly Media, Inc.

Digium. (2015). *Digium Telephony Cards Advantages.* Obtenido de scribd-download.com: http://scribd-download.com/digium-telephony-cards-advantages_58c9f785ee34352a77640ba4_pdf.html

DigiumCards. (2013). *Why Digium Telephony Cards?* Obtenido de www.digium.com: https://www.digium.com/sites/digium/files/digium-telephony-cards-advantages.pdf

Freeswitch. (2017). *The World's First Cross-Platform Scalable FREE Multi-Protocol Soft Switch.* Obtenido de freeswitch.org: https://freeswitch.org/

Freeswitch.org. (2017). *FreeSitch Solutions.* Obtenido de freeswitch.org/confluence: https://freeswitch.org/confluence/display/FREESWITCH/mod_commands

Guillan, E. (1999). *Software para aplicación CTI de primer nivel.* Santiago de Cuba: Universidad de Oriente.

Minessale, A., Collins, M., Schreiber, D., & Chandler, R. (2013). *FreeSWITCH 1.2. Second Edition.* Birmingham: Packt Publishing.

Northforge. (2017). *A Comparison of VOIP Platforms: Asterisk vs. FreeSWITCH.* Obtenido de gonorthforge.com: http://gonorthforge.com/a-comparison-of-voip-platforms-asterisk-vs-freeswitch/

Sampieri, R. H., Collado, C. F., & Lucio, P. B. (2006). *Metodología de la Investigación.* Istapalapa: Mc Graw Hill.

Sangoma. (2016). *Analog Telephony Cards.* Obtenido de Sangoma Everything Connects: https://www.sangoma.com/products/analog-telephony-cards/

Santos, J. (2006). *Centro de Información Telefónico Automatizado para Hospitales utilizando la Integración Computadora Teléfono.* Santiago de Cuba: Universidad de Oriente.

GLOSARIO DE TÉRMINOS

A

ACRI: *Application Controlled Routing For Incoming Calls*

ACRO: *Application Controlled Routing For Outgoing Calls*

C

CDR: *Call Detail Record*

CTI: *Computer Telephony Integration*

D

DLL: *Dynamic Link Library*

DIY: *Do It Yourself*

DTMF: *Dual Tone Multi-Frequency*

F

FOSS: *Free and Open Source Software*

FXO: *Foreign eXchange Office*

FXS: *Foreign eXchange Subscriber*

G

GSM: *Global System for Mobile communications*

H

HTTP: *HyperText Transfer Protocol*

I

IAX: *Inter-Asterisk eXchange protocol*

IDE: *Integrated Development Environment*

IETF: *Internet Engineering Task Force*

IP: *Internet Protocol.*

IVR: *Interactive Voice Response*

M

MPL: *Mozilla Public License*

P

PBX: *Private Branch Exchange*

PIC: *Peripheral Interface Controller*

PSoC: *Programmable System on Chip*

PSTN: *Public Switched Telephone Network*

PWM: *Pulse Width Modulation*

Q

QoS: *Quality of Service*

R

RAM: *Random Access Memory*

RDSI: Red Digital de Servicios Integrados

RTP: *Real-time Transport Protocol*

S

SIP: *Session Initiation Protocol*

SPI: *Serial Peripheral Interface*

SRAM: *Static Random Access Memory*

T

TCP: *Transmission Control Protocol*

TTL: *Transistor-Transistor Logic*

TTS: *Text To Speech*

TDM: Time Division Multiple Access

U

USB: *Universal Serial Bus*

V

VoIP: *Voice over IP*

VRU: *Voice Response Unit*

X

XML: *eXtensible Markup Language*

ANEXO 1 CONFIGURACIÓN DEL PLAN DE LLAMADAS DEL FREESWITCH

Fichero de configuración del plan de llamada o *dialplan* ubicado en la carpeta "conf/dialplan" dentro del directorio raíz del FreeSWITCH.

```
<include>
<contextname="default">

<extensionname="ivr_default">
<conditionfield="destination_number"expression="^(5000)$">
<actionapplication="answer"/>
<actionapplication="sleep"data="2000"/>
<actionapplication="ivr"data="Inicial"/>
</condition>
</extension>

<extensionname="TalkingClockDate">
<conditionfield="destination_number"expression="^5002">
<actionapplication="answer"/>
<actionapplication="sleep"data="1000"/>
<actionapplication="say"data="esCURRENT_DATE_TIMEpronounced${strepoch()}"/>
<actionapplication="hangup"/>
</condition>
</extension>

<extensionname="say_variable">
<conditionfield="destination_number"expression="^(5001)$">
<actionapplication="answer"/>
<actionapplication="sleep"data="2000"/>
<actionapplication="say"data="esnumberpronounced${var_to_say}"/>
</condition>
</extension>

<extensionname="out_call">
<conditionfield="${callIn}"expression="^false$"/>
```

```
<conditionfield="destination_number"expression="^9(\d+)$">
<actionapplication="set_global"data="requestCall=true"/>
<actionapplication="set_global"data="requestNumber=number$1"/>
<actionapplication="bridge"data="portaudio"/>
</condition>
</extension>

<extensionname="Music_Test">
<conditionfield="destination_number"expression="^(9198)$">
<actionapplication="playback"data="C:\files\TERREMOTO.wav"/>
</condition>
</extension>

<extensionname="vmain">
<conditionfield="destination_number"expression="^(5003)$">
<actionapplication="answer"/>
<actionapplication="sleep"data="1000"/>
<actionapplication="voicemail"data="checkdefault${domain_name}"/>
</condition>
</extension>

<extensionname="Local_Extension">
<conditionfield="destination_number"expression="^(|1029|1028|1030)$">
<actionapplication="export"data="dialed_extension=$1"/>
<actionapplication="bind_meta_app"data="1bsexecute_extension::dxXMLfeatures"/>
<actionapplication="bind_meta_app"data="2bsrecord_session::$${recordings_dir}/${calle
r_id_number}.${strftime(%Y-%m-%d-%H-%M-%S)}.wav"/>
<actionapplication="bind_meta_app"data="3bsexecute_extension::cfXMLfeatures"/>
<actionapplication="bind_meta_app"data="4bsexecute_extension::att_xferXMLfeatures"/
>
<actionapplication="set"data="ringback=${us-ring}"/>
<actionapplication="set"data="transfer_ringback=$${hold_music}"/>
<actionapplication="set"data="call_timeout=30"/>
<actionapplication="set"data="hangup_after_bridge=true"/>
<actionapplication="set"data="continue_on_fail=true"/>
<actionapplication="hash"data="insert/${domain_name}-
call_return/${dialed_extension}/${caller_id_number}"/>
<actionapplication="hash"data="insert/${domain_name}-
last_dial_ext/${dialed_extension}/${uuid}"/>
```

<actionapplication="set"data="called_party_callgroup=${user_data(${dialed_extension}@${domain_name}varcallgroup)}"/>
<actionapplication="hash"data="insert/${domain_name}-last_dial_ext/${called_party_callgroup}/${uuid}"/>
<actionapplication="hash"data="insert/${domain_name}-last_dial_ext/global/${uuid}"/>
<actionapplication="hash"data="insert/${domain_name}-last_dial/${called_party_callgroup}/${uuid}"/>
<actionapplication="bridge"data="user/${dialed_extension}@${domain_name}"/>
<actionapplication="answer"/>
<actionapplication="sleep"data="1000"/>
<actionapplication="bridge"data="loopback/app=voicemail:default${domain_name}${dialed_extension}"/>
</condition>
</extension>
</context>
</include>

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Dominguez Diaz Guido Daniel**, con C.C: # **1203841638** autor/a del trabajo de titulación: **Sistema de respuesta de voz interactiva basado en FREESWITCH para centros de información automatizados** previo a la obtención del título de **Magíster en Telecomunicaciones** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, a los 21 días del mes de agosto del año 2017

f. _____

Nombre: **Dominguez Diaz Guido Daniel**

C.C: **1203841638**

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN			
TÍTULO Y SUBTÍTULO:	Sistema de respuesta de voz interactiva basado en FREESWITCH para centros de información automatizados		
AUTOR(ES)	Dominguez Diaz Guido Daniel		
REVISOR(ES)/TUTOR	MSc. Orlando Philco Asqui; MSc. Luis Córdova Rivadeneira / MSc. Manuel Romero Paz		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Sistema de Posgrado		
PROGRAMA:	Maestría en Telecomunicaciones		
TITULO OBTENIDO:	Magister en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	**21 de Agosto de 2017**	**No. DE PÁGINAS:**	**54**
ÁREAS TEMÁTICAS:	Sistemas IVR, Aplicaciones CTI, Asterisk, FreeSWITCH, Plan de llamadas, Tarjetas Sangoma, Entorno de Desarrollo Integrado (IDE)		
PALABRAS CLAVES/ KEYWORDS:	IVR, CTI, FreeSWITCH, X-Lite, IDE, Asterisk		
RESUMEN/ABSTRACT: En este trabajo se presenta una propuesta de un sistema de Respuesta de Voz Interactiva (IVR) usando software libre para la implementación de centros de información automatizados en beneficio del proceso de informatización de la sociedad. Se describen las generalidades de la integración computadora teléfono (CTI), los softwares existentes que permiten la creación de sistemas IVR y se fundamenta el uso del FreeSWITCH como mejor alternativa. Se explican las distintas configuraciones realizadas en el FreeSWITCH para el correcto funcionamiento del menú IVR y otras aplicaciones requeridas. Se exponen los criterios que avalan la selección del software X-Lite como cliente. Se seleccionan las tarjetas de interfaz con la línea telefónica más apropiadas para la aplicación considerando costo y máxima compatibilidad con FreeSWITCH. Finalmente, se muestra el diseño de la aplicación de control del sistema, implementada con el Entorno de Desarrollo Integrado (IDE) Visual Studio 2015, describiéndose sus funcionalidades a través de la presentación de la interfaz gráfica.			
ADJUNTO PDF:	☒ SI	☐ NO	
CONTACTO CON AUTOR/ES:	**Teléfono:** +593-999560723	**E-mail:** guidodaniel@hotmail.es	
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre:** Romero Paz Manuel de Jesús		
	Teléfono: +593-994606932		
	E-mail: manuel.romero@cu.ucsg.edu.ec		
SECCIÓN PARA USO DE BIBLIOTECA			
Nº. DE REGISTRO (en base a datos):			
Nº. DE CLASIFICACIÓN:			
DIRECCIÓN URL (tesis en la web):			